NAME

update — Process pending events and idle callbacks

SYNOPSIS

update ?**idletasks**?

DESCRIPTION

This command is used to bring the application “up to date” by entering the event loop repeatedly until all pending events (including idle callbacks) have been processed.

If the **idletasks** keyword is specified as an argument to the command, then no new events or errors are processed; only idle callbacks are invoked. This causes operations that are normally deferred, such as display updates and window layout calculations, to be performed immediately.

The **update idletasks** command is useful in scripts where changes have been made to the application's state and you want those changes to appear on the display immediately, rather than waiting for the script to complete. Most display updates are performed as idle callbacks, so **update idletasks** will cause them to run. However, there are some kinds of updates that only happen in response to events, such as those triggered by window size changes; these updates will not occur in **update idletasks**.

The **update** command with no options is useful in scripts where you are performing a long-running computation but you still want the application to respond to events such as user interactions; if you occasionally call **update** then user input will be processed during the next call to **update**.

EXAMPLE

Run computations for about a second and then finish:

```
set x 1000
set done 0
after 1000 set done 1
while {!$done} {
    # A very silly example!
    set x [expr {log($x) ** 2.8}]

    # Test to see if our time-limit has been hit.  This would
    # also give a chance for serving network sockets and, if
    # the Tk package is loaded, updating a user interface.
    update
}
```

SEE ALSO

after, interp

KEYWORDS

asynchronous I/O, event, flush, handler, idle, update